Y. Ioannidis et al. (eds.): EDBT 2006, LNCS 3896, pp. 1147–1150, 2006.

TQuEST: Threshold Query Execution for Large Sets of Time Series

Johannes Aßfalg, Hans-Peter Kriegel, Peer Kröger, Peter Kunath, Alexey Pryakhin, Matthias Renz

Institute for Computer Science, University of Munich
{assfalg,kriegel,kroegerp,kunath,pryakhin,renz}@dbs.ifi.lmu.de

Abstract. Effective and efficient data mining in time series databases is essential in many application domains as for instance in financial analysis, medicine, meteorology, and environmental observation. In particular, temporal dependencies between time series are of capital importance for these applications. In this paper, we present TQuEST, a powerful query processor for time series databases. TQuEST supports a novel but very useful class of queries which we call *threshold queries*. Threshold queries enable searches for time series whose values are above a user defined threshold at certain time intervals. Example queries are "report all ozone curves which are above their daily mean value at the same time as a given temperature curve exceeds $28°C$" or "report all blood value curves from patients whose values exceed a certain threshold one hour after the new medication was taken". TQuEST is based on a novel representation of time series which allows the query processor to access only the relevant parts of the time series. This enables an efficient execution of threshold queries. In particular, queries can be readjusted with interactive response times.

1 Introduction

In this paper, we present TQuEST, a powerful analysis tool for time series databases which supports a novel but very important query type which we call *threshold query*. Given a query time series q, a threshold τ, and a time series database DB, a threshold query $TQ_{DB}(q, \tau)$ returns the time series $p \in DB$ that has the most similar sequence of intervals of values above τ. In other words, each time series $o \in DB \cup \{q\}$ is transformed into a sequence of disjoint time intervals containing only those values of o that are (strictly) above τ. Then, a threshold query returns for a given query object q the object $p \in DB$ having the most similar sequence of time intervals. Let us note that the exact values of the time series are not considered, rather we are only interested in whether the time series is above or below a given threshold τ. The transformation of the time series into interval sequences is depicted in Figure 1.

Our new query type can for example be applied to pharmacological time series like blood parameters after drug treatment or biological data like gene expression profiles. Another possible application for our tool is the analysis of

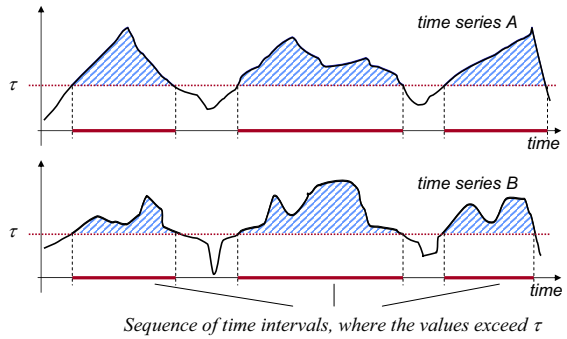


Fig. 1. Transformation of time series into sequences of time intervals.

environmental air pollution which has gained rapidly increasing attention by many European research projects in recent years. For example, German state offices for environmental protection maintain more than 100 million time series, each representing the daily course of air pollution parameters[1]. It is important to know which parameters nearly simultaneously exceed their legal threshold.

A lot of work on similarity search in time series databases has been published. The proposed methods mainly differ in the representation of the time series, a survey is given in [1]. Standard techniques for dimension reduction include Discrete Fourier Transform (e.g. [2]), Discrete Wavelet Transform (e.g. [3]), Piecewise Aggregate Approximation (e.g. [4]), and Chebyshev Polynomials [5]. However, all techniques which are based on dimension reduction cannot be applied to threshold queries because necessary temporal information is lost.

An effective and efficient processing of queries like "return all ozone time series which exceed the threshold $\tau_1 = 50\mu g/m^3$ at a similar time as the temperature reaches the threshold $\tau_2 = 25°C$" is of high importance but not supported by the above mentioned techniques. Such a query type has to support the exploration of time series based on user-defined amplitude spectrums. TQuEST not only meets these prerequisites but also enables the user to interactively adjust the query threshold.

2 Time Series Representation

As time series objects may be very complex, i.e. contain lots of measurements, we need a data representation of time series objects which allows us to process threshold queries very efficiently. An efficient processing enables interactive query response times, so that the query can be readjusted w.r.t. the last results without causing high response times. In the following, we assume that a time series is described by a sequence of connected segments and each segment denotes the interpolated time series course between a pair of subsequent time series values (measurements).

[1] We thank U. Böllmann and M. Meindl for providing us with real-world datasets from the Bavarian State Office for Environmental Protection, Augsburg, Germany.

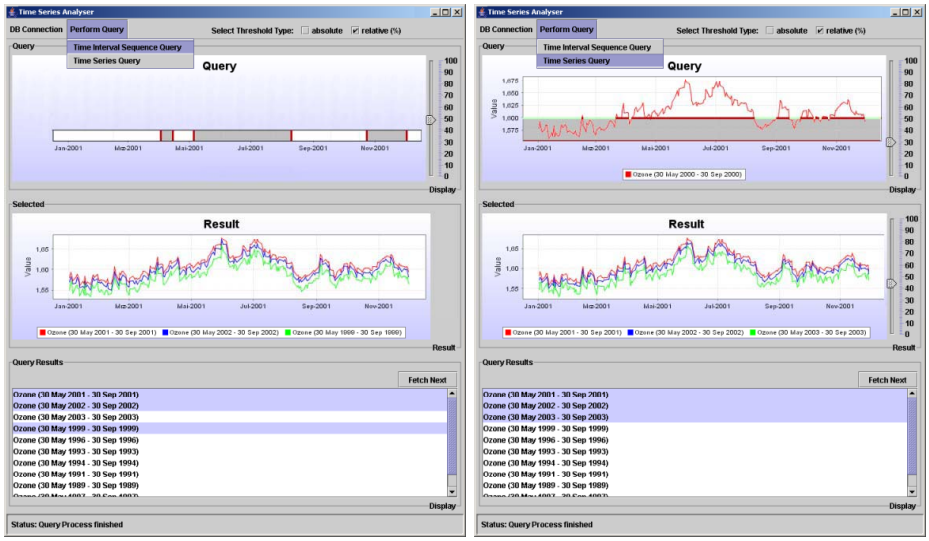
The query processor is based on a novel data representation of time series. Assume a threshold query with a threshold τ is given. At query time, the query processor requires for each time series to derive the relevant time intervals, i.e. the time frames where the time series course is above the specified threshold τ. Obviously, those segments of the time series which cross the query threshold τ suffice to determine the desired time frames. This observation can be used as follows: we decompose the time series into trapezoids where the upper and lower edge is parallel to the time axis, the left side is bounded by an increasing time series segment and the right side is bounded by a decreasing segment. For a certain range of thresholds and a certain range of time, each trapezoid represents all time frames where the time series course is above the threshold. The trapezoids can be described very compactly and can be efficiently accessed by means of a spatial access method (e.g. [6]). The key point of our proposed time series representation is that we do not need to access the complete time-series data at query time. Instead only partial information of the time-series objects suffices to report the results.

3 Threshold Query Processor

In this section, we present TQuEST, a JAVA 1.5 application, which supports effective threshold queries efficiently. At first the user has to select the desired *threshold type*. In case of an *Absolute Threshold* τ, all underlying calculations are based on absolute time series values. Our tool also handles *Relative Thresholds* where τ is given relative to the maximum and the minimum values of the time series. This query mode requires that all time series are also represented in a normalized form. In many application fields, a number of parameters is observed at the same time. Quite often these measurements yield time series with totally different ranges of values. By using relative thresholds, our tool is nonetheless able to detect correlations between certain observed parameters (for example between temperature and ozone concentration).

TQuEST supports two major types of queries: In the *Time Interval Sequence Query* mode, the user can specify a sequence of time intervals on the time bar and a threshold for the query. Our software then retrieves time series that match these parameters best (cf. Figure 2(a)). The *Time Series Query* mode uses a given time series as a query object. Here, the user has to provide a threshold τ_1 for the query time series and a threshold τ_2 for the objects to retrieve. Our software then searches for time series that exceed τ_2 during the same (or similar) time intervals as the query time series exceeds τ_1 (cf. Figure 2(b)).

In both query modes, our tool ranks the resulting time series according to the similarity between the specified time intervals and the time intervals in which the result time series exceed the threshold. Depending on the available meta-data for each time series our tool can be configured to display any combination of additional attributes associated with a specific time series (e.g. patient ID, observed parameter, geographical location of the corresponding sensor station, etc.). To get a quick visual impression of the results, multiple time series can be selected for the graphical display in the middle area of the query GUI. Finally,



(a) Time interval sequence query (b) Time series query

Fig. 2. Query interface

by clicking the 'Fetch Next' button, the user can retrieve the next results from the database w.r.t. the ranking order.

We demonstrate on real-world data that our prototype TQuEST is very useful. For example, we discovered relationships between meteorological data (e.g. air humidity) and air pollution attributes (e.g. particulate matter). Furthermore we can show that our tool can handle complex threshold queries in adequate time.

References

1. Keogh, E., Chakrabati, K., Mehrotra, S., Pazzani, M.: "Locally Adaptive Dimensionality Reduction for Indexing Large Time Series Databases". In: Proc. ACM SIGMOD Int. Conf. on Management of Data, Santa Barbara, CA. (2001)
2. Agrawal, R., Faloutsos, C., Swami, A.: "Efficient Similarity Search in Sequence Databases". In: Proc. 4th Conf. on Foundations of Data Organization and Algorithms. (1993)
3. Chan, K., Fu, W.: "Efficient Time Series Matching by Wavelets". In: Proc. 15th Int. Conf. on Data Engineering (ICDE'99), Sydney, Australia. (1999)
4. Yi, B.K., Faloutsos, C.: "Fast Time Sequence Indexing for Arbitrary Lp Norms". In: Proc. 26th Int. Conf. on Very Large Databases (VLDB'00), Cairo, Egypt. (2000)
5. Cai, Y., Ng, R.: "Index Spatio-Temporal Trajectories with Chebyshev Polynomials". In: Proc. ACM SIGMOD Int. Conf. on Management of Data, Paris, France). (2004)
6. Beckmann, N., Kriegel, H.P., Seeger, B., Schneider, R.: "The R*-tree: An Efficient and Robust Access Method for Points and Rectangles". In: Proc. ACM SIGMOD Int. Conf. on Management of Data, Atlantic City, NJ. (1990)